VLSI Decoder Architecture for High Throughput, Variable Block-size and Multi-rate LDPC Codes

Yang Sun, Marjan Karkooti and Joseph R. Cavallaro
Department of Electrical and Computer Engineering
Rice University, Houston, TX 77005
Email: {ysun, marjan, cavallar}@rice.edu

***Abstract*— A low-density parity-check (LDPC) decoder architecture that supports variable block sizes and multiple code rates is presented. The proposed architecture is based on the structured quasi-cyclic (QC-LDPC) codes whose performance compares favorably with that of randomly constructed LDPC codes for short to moderate block sizes. The main contribution of this work is to address the variable block-size and multi-rate decoder hardware complexity that stems from the irregular LDPC codes. The overall decoder, which was synthesized, placed and routed on TSMC 0.13-micron CMOS technology with a core area of 4.5 square millimeters, supports variable code lengths from 360 to 4200 bits and multiple code rates between 1/4 and 9/10. The average throughput can achieve 1 Gbps at 2.2 dB SNR.**

I. Introduction

Low-density parity-check (LDPC) codes have received tremendous attention in the coding community because of their excellent error correction capability and near-capacity performance. Some randomly constructed LDPC codes, measured in Bit Error Rate (BER), come very close to the Shannon limit for the AWGN channel (within 0.05 dB) with iterative decoding and very long block sizes (on the order of 10^6 to 10^7). However, for many practical applications (e.g. packet-based communication systems), shorter and variable block-size LDPC codes with good Frame Error Rate (FER) performance are desired. Communications in packet-based wireless networks usually involve a large per-frame overhead including both the physical (PHY) layer and MAC layer headers. As a result, the design for a reliable wireless link often faces a trade-off between channel utilization (frame size) and error correction capability. One solution is to use adaptive burst profiles in which transmission parameters relevant to modulation and coding may be assigned dynamically on a burst-by-burst basis. Therefore, LDPC codes with variable block lengths and multiple code rates for different quality-of-service under various channel conditions are highly desired.

In the recent literature, there are many LDPC decoder architectures but few of them support variable block-size and muti-rate decoding. For example, in [1] a 1 Gbps 1024-bit, rate $1/2$ LDPC decoder has been implemented. However this architecture just supports one particular LDPC code by wiring the whole Tanner graph into hardware. In [2], a code rate programmable LDPC decoder is proposed, but the code length is still fixed to 2048 bit for simple VLSI implementation. In [3], a LDPC decoder that supports three block sizes and four code rates is designed by storing 12 different parity check matrices on-chip. As we can see, the main design challenge for supporting variable block sizes and multiple code rates stems from the random or unstructured nature of the LDPC codes. Generally support for different block sizes of LDPC codes would require different hardware architectures. To address this problem, we propose a generalized decoder architecture based on the quasi-cyclic LDPC (QC-LDPC) codes that can support a wider range of block sizes and code rates at a low hardware requirement.

II. Structured QC-LDPC codes

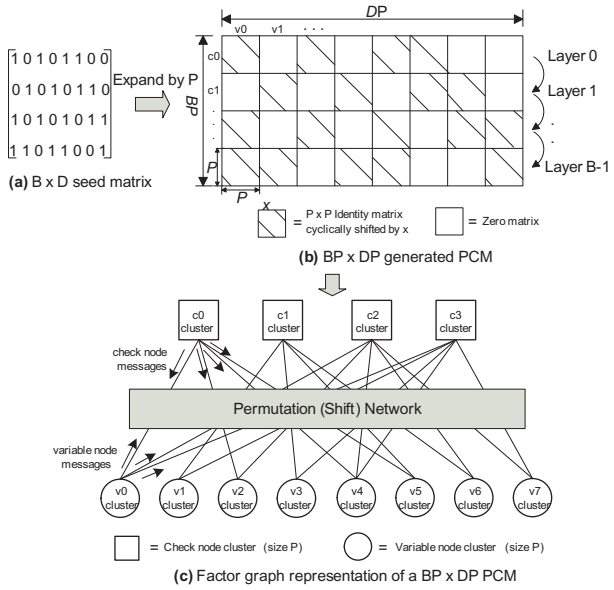

Fig. 1. Parity check matrix and its factor graph representation

To balance the implementation complexity and the decoding throughput, a structured QC-LDPC code was proposed in [4] recently for modern wireless communication systems including but not limited to IEEE $802.16e$ and IEEE $802.11n$.

As shown in Fig. 1(a)(b), for a QC-LDPC code, the parity check matrix (PCM) is constructed from a $B \times D$ seed matrix by replacing each '1' in the seed matrix with a $P \times P$ cyclically shifted identity sub-matrix, where P is an expansion factor. A corresponding Tanner factor graph representation of this $BP \times DP$ generated PCM is shown in Fig. 1(c). It divides

1-4244-0921-7/07 $25.00 © 2007 IEEE.

the variable nodes and the check nodes into clusters of size P such that if there exists an edge between variable and check clusters, then it means P variable nodes connect to P check nodes via a permutation (cyclic shift) network.

Generally, support for different block sizes and code rates implies usage of multiple PCMs. Storing all the PCMs on-chip is almost impractical and expensive. In this work, we utilize the expansion factor P so that only one parity check matrix needs to be stored for a given seed matrix. Denote P_0 as the largest possible expansion factor for a given seed matrix, we can construct a QC-LDPC code of $B \times D$ array of $P_0 \times P_0$ sub-matrices, which are either zero matrices or cyclically shifted identity matrices. The corresponding shift values denoted as $m(P_0, i, j)$ are stored on-chip. For all the other expansion factors P_x, the shifted values are derived from $m(P_0, i, j)$ by:

$$m(P_x, i, j) = \lfloor \frac{m(P_0, i, j) \cdot P_x}{P_0} \rfloor. \quad (1)$$

Obviously, with the help of expansion factor P_x, ($P_x < P_0$), we are able to generate different size PCMs from the same seed matrix to support different size codes. However, to support different code rates, different seed matrices as well as the shift values associated with each seed matrix must be constructed. Table I presents the seed matrices needed to support different code rate requirements. Each seed matrix can be constructed by an algebraic construction method proposed by Tanner in [4].

TABLE I
CODE RATE VERSUS SEED MATRIX

$Rate$	H_{seed}	$Rate$	H_{seed}	$Rate$	H_{seed}
1/4	18×24	3/5	10×25	5/6	4×24
1/3	16×24	2/3	8×24	7/8	3×24
2/5	15×25	3/4	6×24	8/9	3×27
1/2	12×24	4/5	5×25	9/10	3×30

III. LDPC DECODER HARDWARE ARCHITECTURE

A. Layered partially parallel soft decoding algorithm

A good tradeoff between design complexity and decoding throughput is partially parallel decoding by grouping a certain number of variable and check nodes into a cluster for parallel processing. Furthermore, the layered decoding algorithm [5] can be applied to improve the decoding convergence time by a factor of two and hence increases the throughput by 2X.

The structured QC-LDPC code makes it effectively suitable for efficient VLSI implementation by significantly simplifying the memory access and message passing. As shown in Fig. 1(b), the PCM can be viewed as a group of concatenated horizontal layers, where the column weight is at most 1 in each layer due to the cyclic shift structure. The belief propagation algorithm is repeated for each horizontal layer and the updated APP (*a posteriori* probability) messages are passed between layers. Let R_{ij} denote the check node LLR (Log-likelihood ratios) messages sent from the check node i to the variable node j. Let $L(q_{ij})$ denote the variable node LLR messages sent from the variable node j to the check node i. Let $L(q_j)$ ($j = 1, \ldots, N$) represent the APP messages for all the variable nodes (coded bits) which are initialized with the channel messages (assuming BPSK on AWGN channel) for each code bit j by $2r_j/\sigma^2$, where σ^2 is the noise variance and r_j is the received value. For each variable node j inside the current horizontal layer, messages $L(q_{ij})$ that correspond to a particular check equation i are computed according to:

$$L(q_{ij}) = L(q_j) - R_{ij}. \quad (2)$$

For each check node i, messages R_{ij}, corresponding to all variable nodes j that participate in a particular parity-check equation, are computed according to:

$$R_{ij} = \prod_{j' \in N(i) \setminus \{j\}} \text{sign}\left(L(q_{ij'})\right) \Psi \left[\sum_{j' \in N(i) \setminus \{j\}} \Psi\left(L(q_{ij'})\right) \right], \quad (3)$$

where $N(i)$ is the set of all variable nodes from parity-check equation i, and $\Psi(x) = -\log\left[\tanh\left(\frac{|x|}{2}\right)\right]$. The APP messages in the current horizontal layer are updated by:

$$L(q_j) = L(q_{ij}) + R_{ij}. \quad (4)$$

For QC-LDPC codes, the parity check matrix can be viewed as a $B_{row\_cluster} \times D_{column\_cluster}$ structure by grouping variable nodes and check nodes into clusters of size P. Now let i and j denote the row cluster index and the column cluster index, a layered partially parallel decoding algorithm is given by:

for iter = $0 : max\ iteration - 1$
  for layer (row cluster) $i = 0 : B - 1$
    for column cluster $j = 0 : D - 1$
      if $PCM_{i,j}$ *is a non-zero sub-matrix* {
        Read a cluster of APP data $L(q_j)$ *from APP memory*
        Read a cluster of Check data R_{ij} *from Check memory*
        Calculate shift value from (1) *and permute APP data*
        Calculate equation $(2)(3)(4)$
        Update new APP and Check data to memory
      }

where the decoding will stop whenever all the parity check constraints are satisfied or the max number of iterations is reached.

B. Min-sum algorithm and fixed-point implementation

The belief propagation algorithm [6] is the most powerful iterative soft decoding algorithm for LDPC codes. But due to its high design complexity in (3), many implementations for decoding LDPC codes are based on the modified (normalized or offset) min-sum algorithm because of its satisfactory performance and simple implementation [7]. By applying the offset min-sum algorithm, equation (3) is reduced to:

$$R_{ij} \approx \prod_{j' \in N(i) \setminus \{j\}} \text{sign}\left(L(q_{ij'})\right) \times \max\left(\min_{j' \in N(i) \setminus \{j\}} |L(q_{ij'})| - \beta, 0 \right) \quad (3')$$

For logic circuit design with finite precision, we consider the received values to be quantized in the range of $[-Z, Z]$ and represented by W quantization bits. With a properly chosen offset value β and Z, a 6-bit quantized min-sum algorithm exhibits only about 0.1 dB of degradation in performance compared with the unquantized standard BP algorithm [7].

C. Partially parallel decoder architecture

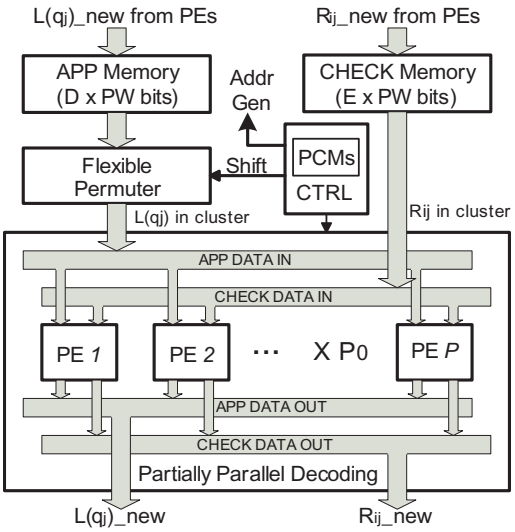

Fig. 2. Top level decoder architecture

Fig. 2 shows the block diagram of the decoder architecture based on the layered partially parallel decoding algorithm. In each sub-iteration, a cluster of APP messages and check messages are fetched from APP and Check memory, and then the APP messages are passed through a flexible permuter to be routed to the correct Processing Engines (PEs) for updating new APP messages and check messages. The PEs are the central processing units of the architecture that are responsible for updating messages based on $(2)(3^{'})(4)$. The number of PEs determines the parallelism factor of the design. For a certain block-size code, only P_x PEs are working while the rests are in a power saving mode. As shown in Fig. 3, the PE inputs wr elements of $L(q_j)$ and R_{ij}, where wr is the number of nonzero values in each row of the PCM. $L(q_{ij})$ is calculated based on (2). The sign and magnitude of $L(q_{ij})$ are processed based on $(3^{'})$ to generate new R_{ij}. Then the $L(q_{ij})$ are added to the R_{ij} to generate new $L(q_j)$ (w_r of them) based on (4). The outputs ($L(q_j)$ and R_{ij}) of all the P_x PEs are concatenated and stored in one address of the APP and Check memories. For each layer's sub-iteration, it takes about $2wr$ clock cycles to process, so the decoding throughput is:

$$Throughput \approx \frac{D \times P_x \times R \times fclk_{max}}{2 \times E \times iterations}$$

where R is the code rate and E is the total number of edges between all variable nodes and check nodes in the seed matrix. Clearly, the throughput would be linearly proportional to the expansion factor P_x for a given seed matrix.

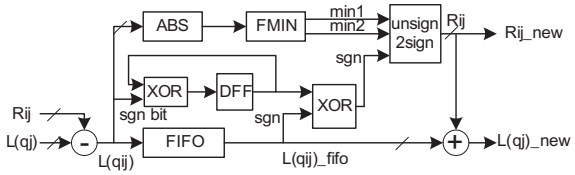

Fig. 3. Processing Engine (PE)

D. Flexible permuter

One of the main challenges of the LDPC decoder architecture is the permuter (π) design that is responsible for routing the messages between variable nodes and check nodes. However for QC-LDPC codes, the permuter is just a barrel shifter network (size-P) for cyclically shifting the node messages to the correct PEs. Fig. 4 gives an example of a size-4 barrel shifter network. The hardware design complexity of this type of network is $O(P\lceil \log_2 P \rceil)$ as compared to $O(P^2)$ for the directly connected network. For large size P (e.g. 128), the barrel shifter network needs to be partitioned into multiple pipeline stages for high speed VLSI implementation.

Traditionally a de-permuter (π^{-1}) would be needed to permute the shuffled data back and save it to memory, which would occupy a significant portion of the chip area [2]. However, due to the cyclic shift property of the QC-LDPC codes, no de-permuter is needed. We can just store the shuffled data back to memory and for the next iteration we should then shift this "shuffled data" by an incremental value $\Delta = (shift_n - shift_{n-1})\ mod\ P$.

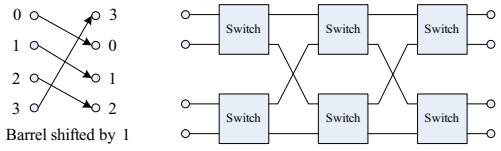

Fig. 4. A 4×4 Barrel shifter network

E. Pipelined decoding for higher throughput

The decoding throughput can be further improved by overlapping the decoding of two layers using a pipelined method. The decoding of each layer of the parity check matrix is

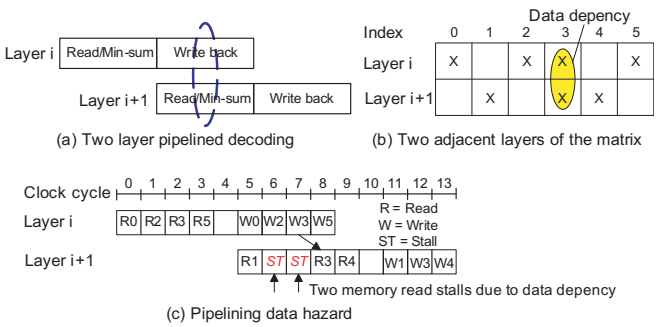

Fig. 5. Pipelined decoding

performed in two stages: 1) Memory read and min-sum calculation and 2) Memory write back. However, due to the possible data dependence between two consecutive layers (there is no data dependency inside each layer because the column weight is at most 1 in each layer), a pipelining data hazard might occur. Fig. 5 shows an example of pipelined decoding. In Fig. 5(c), at clock cycle 6, layer $(i+1)$ is trying to access APP memory address 3 which will not be updated by layer i until clock cycle 7, hence two pipeline stalls need to be inserted. Moreover, a horizontal rescheduling algorithm can also be applied to help reduce pipeline stalls. For example, in Fig. 5, layer $(i+1)$'s reading can be rescheduled from the original sequence 1-3-4 to 1-4-3 to reduce pipeline stalls. This way, the decoding throughput will be increased to

$$Pipelined\ Throughput \approx \frac{D \times P_x \times R \times fclk_{max}}{E \times iterations}$$

IV. Physical VLSI Design

A flexible LDPC decoder which supports variable block sizes from 360 to 4200 bits in fine steps, where the step size can be 24 (at rate $1/4$, $1/3$, $1/2$, $2/3$, $3/4$, $5/6$ and $7/8$), or 25 (at rate $2/5$, $3/5$ and $4/5$), or 27 (at rate $8/9$), or 30 (at rate $9/10$), was described in Verilog HDL. Layout was generated for a TSMC $0.13\mu m$ CMOS technology as shown in Fig. 6

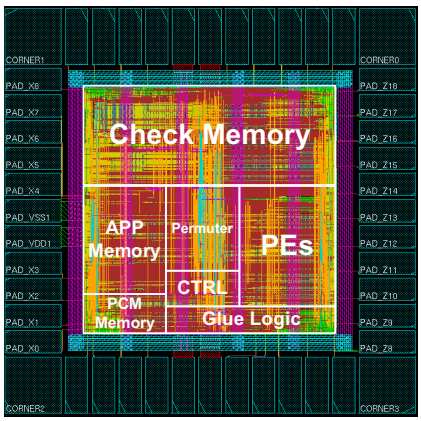

Fig. 6. Flexible LDPC decoder VLSI layout ($0.13\mu m$)

V. Performance Analysis and Comparison

Fig. 7 shows the FER performance and compares the two cases that also exist in the IEEE $802.11n$ (WWiSE Proposal) codes. Table II compares this decoder with the state-of-the-art LDPC decoders of [1] and [2]. As we can see, the proposed decoder shows significant performance in throughput, flexibility, area and power.

VI. Conclusion

A VLSI decoder architecture that supports variable block-size and multi-rate LDPC codes has been presented. By utilizing structured QC-LDPC codes, we proposed a pipelined partially parallel decoding algorithm which is well suited for VLSI implementation. The decoder has been placed and routed

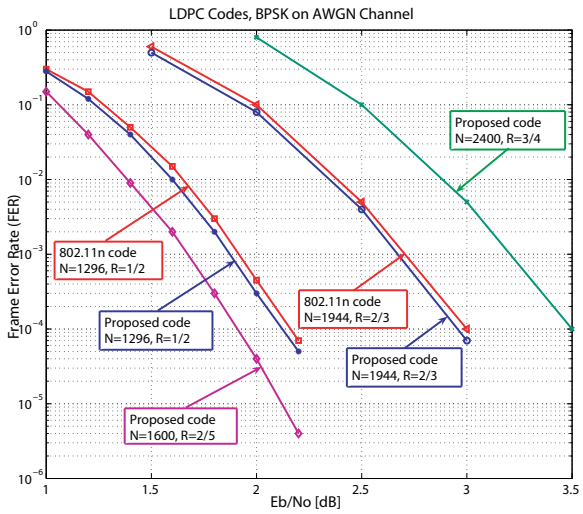

Fig. 7. FER performance comparison with IEEE $802.11n$ codes

TABLE II
Comparison of Proposed Decoder with Existing LDPC Decoders

	Proposed Decoder	Blanksby [1]	Mansour [2]
Throughput	1.0 Gbps@2.2dB	1.0 Gbps	1.3Gbps@2.2dB
Area	4.5 mm^2	52.5mm^2	14.3 mm^2
Frequency	350 MHz	64 MHz	125 MHz
Power	740 mW	690 mW	787 mW
Block size	360 to 4200 bit	1024 bit fixed	2048 bit fixed
Code Rate	$1/4 : 9/10$	$1/2$ fixed	$1/16 : 14/16$
Technology	$0.13\mu m, 1.2V$	$0.16\mu m, 1.5V$	$0.18\mu m, 1.8V$

using TSMC $0.13\ \mu m$, $1.2V$, eight metal layers CMOS technology. The decoder can support high throughput decoding, for example, 1 Gbps at 2.2 dB SNR, at less area.

VII. Acknowledgement

This work was supported in part by Nokia and by NSF under grants CCF-0541363, CNS-0551692, and CNS-0619767.

References

[1] A.J. Blanksby and C.J. Howland, "A 690-mW 1-Gb/s 1024-b, rate-1/2 low-density parity-check code decoder," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 3, pp. 404–412, 2002.

[2] M.M. Mansour and N.R. Shanbhag, "A 640-Mb/s 2048-Bit Programmable LDPC Decoder Chip," *IEEE Journal of Solid-State Circuits*, vol. 41, pp. 684–698, March 2006.

[3] M. Karkooti, P. Radosavljevic, and J. R. Cavallaro, "Configurable, High Throughput, Irregular LDPC Decoder Architecture:Tradeoff Analysis and Implementation," *IEEE 17th International Conference on Application-specific Systems, Architectures and Processors*, pp. 360–367, Sep. 2006.

[4] R.M. Tanner, D. Sridhara, A. Sridharan, T.E. Fuja, and D.J. Costello Jr., "LDPC block and convolutional codes based on circulant matrices," *IEEE Transactions on Information Theory*, vol. 50, no. 12, pp. 2966–2984, 2004.

[5] M. M. Mansour and N. R. Shanbhag, "High-throughput LDPC decoders," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, pp. 976–996, Dec. 2003.

[6] R. Gallager, "Low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 8, pp. 21–28, Jan. 1962.

[7] J. Chen, A. Dholakia, E. Eleftheriou, M. Fossorier and X. Hu, "Reduced-Complexity Decoding of LDPC Codes," *IEEE Transactions on Communications*, vol. 53, pp. 1232–1232, 2005.